Visualized Music Expression
in an Object-Oriented Environment

Rumi Hiraga
Tsukuba College of Technology

Shigeru Igarashi
University of Tsukuba

Yohsei Matsuzawa
University of Tsukuba

Abstract

In pursuit of expressive performance by a computer, Psyche project has developed several systems including real-time accompaniments or expression synthesis. A visualization system is a tool to analyze and edit music expression which is shown based on music structure. The visualized expression gives more information than a picture which simply shows values of volume and speed because music performance relates closely to the music structure. Also the interactivity of the system is useful to Psycho's performance synthesis system. Using the visualization system, the music could be performed with the feedback of sights and modifying the expression is rather WYSIWYG way than giving parameter values by hand. Internally, music structure (sentence, phrase, and motif) is realized by classes which refer extended MIDI data.

1 Introduction

With the spread of multimedia products these days, there are many sequencers or editors which have GUIs for music manipulation. The visualization of music let more people cope with computer music in various respects. Also in the research area, GUI is now more than nice to have for music systems. Since GUI is originally attached to object-oriented concept [2], many systems are developed with an object-oriented approach. Diphone [1] and Mode [6] are examples.

A system for visualizing music expression of Psyche project intends to analyze and edit music expression. The expression is shown based on a music structure which leads the easiness of the analysis of performance. The analysis may yield some general performance rules and special performances which correspond to originality. Psyche's performance synthesis system, which generates the whole performance with portions of human performance as seeds by applying expression unfolding rules, utilizes the visualization tools for obtaining rules [4]. Also the interactivity of the system is useful when the modification of performance parameters, either volume or length of a note, is required. Using the visualization system, the music could be performed with the feedback of sights, and the interactivity makes the system more than a classical way of analyzing the performance data as in Widmer's system [8].

There are two types of pictures, to display detailed information of a phrase (note length, note volume, pitch, beat length, and mean beat length are shown) and to display numbers of phrases, synchronizing a performance, with the information of beat length or beat volume. Internally, music structure is realized by classes which refer extended MIDI data. The information about similarity or transformation, such as inversion, among structures is also realized as objects.

In Section 2, two types of visualization systems are introduced. Then the interactivity of the system is described in Section 3. The object-oriented approach of the system construction is explained in Section 4.

2 Expression Visualization

Psyche has been pursuing automatic computer performance which is expressive itself and could be a good partner of human players. As systems use the previously prepared score data, Psyche has to solve some problems in converting score data into performance data as mentioned in ROBATO [6]. With that these and the problem in mind, some music visualization systems have been developed. One of item is an editor to show and manipulate the expression by the elasticity of the length of each beat is a motif in five types of expression curves [3].

Currently in Psycho, expression is synthesized by applying performance rules to a specified portion of the performance by a human player. Performance by actual human players is realized both by their own style (individuality) and common style (performance role). For those who can play instruments will be able to think of how one
plays on some specific part of the music. The importance is rules which are used in the synthesis system should be systematically described and are common to many players. Thus one of the purposes of the visualization tool is to get performance rule, in other words the commonality of performance, by analyzing the human performance.

While performers' styles are recognizable to listeners, it won't be easy to describe subjectively which performance way makes the difference of each player. If the styles are specified at least qualitatively, it will help to synthesize expressive performance. The derivation of players' styles is another purpose of the visualization system.

With these purposes and the general requirement for GUI tools, say the representation should be along with the way people think as described in [7] in mind, two types of visualization tools were developed. The first is to display detailed information of a phrase as in Figure 1. The figure shows the play of the first four measures (a phrase) of Mazurka No. 7 by Chopin by three different players. Performance by player A is an output of compiled music score data, player B is a famous pianist in the world, and performance by player C is a synthesized one by Psycho's expression synthesis system with rules. The phrase starts from the top of the circle, then moves on clockwise. Each painted fan shape shows note length in its angle, note volume in its radius, and pitch in color respectively. The thin-lined arcs show beat length and the mean beat length in the phrase. The other is to display numbers of phrases which synchronizes a performance with the information of beat length or volume as in radii of fan shapes (Figure 2). It shows the beat length of the first four phrases of Mazurka No. 7 by the player B in Figure 1.

The former one is usable to get the common way of performance when comparing performances by several players. Also the performance style could be checked with the picture. With the rich information, more music-oriented information, such as leaps shown as the overlapping of contiguous fan shapes, could be derived besides just length and volume. While the latter shows the larger figure of a piece, it could be used to see how music structures which are supposed to have formative relationships are played. Currently phrases are given to the system, in spite of the fact that phrasing is not detected uniquely. The second one may help to find whether the given music structure is appropriate.

Hiraga et al.

3 Editing Performance Interactively

The visualization system which shows a snake-like figure (Figure 2) has a feature of the interactive value change in radius of fan shapes (beat length or beat volume). The interactiveness is useful to generate performance and makes system more than a performance analysis tool. Though it is easy for human to perceive which part of the performance should be changed by listening to the play, currently there is no good user interface to change the expression by What You Hear Is What You Get. By mapping the performance onto a picture, users can try to modify performance several times with the visualized feedback. Needless to say, it is far much better and cost effective than giving performance parameter values in MIDI data by hand.

One of the users of the system in Psycho's music expression synthesis. The synthesis system generates the whole performance with portions of human performance as seeds by applying expression unfolding rules. The first step to generate the whole performance is done by applying rules. Then it is shown on the system with the play at the same time. Generally, the generated performance should be tuned to be aesthetically expressive music. The tuning includes to decrease or to increase values in some beats.

The system has some functions to aid the tuning by considering other systems of Psycho. Expression curves described in section 2 are used as templates to the length of beat for performance tuning. Users specify a motif to be tuned, which type of curve to apply, and arguments for the shape of the curve, then a motif's values of length of each beat is changed.

Another useful function is modification effect. It is to apply the modification to other music structures which have some relationships with the modified one. If a motif M2's an estimato motif of a motif M1, or a motif M3 is a sequence motif of M1 then the change on M1 should affect M2 and M3 in appropriate ways. Some rules are described in the synthesis system in the way to affect modification on a music structure to other music structures. An example rule is that an estimato motif is played softer than a motif previously appeared. Thus when volume in a beat in M1 is changed to softer, then M2's volume is made softer according to a change in M1. The effect is decided both on the rule's parameter values to define how much softer the estimato motif should be and the interactively changed rate. In the next section, the way to give the relationship is shown.

In one of the possible future extension, the performance synthesis system is accessed through
the visualization system. Then the dynamic selection/emission of rules will be added which is useful to synthesize performance.

4 Object-Oriented Approach

The object-oriented approach was taken because the effectiveness of the development for many temporary project stuff (namely students). Also the GUI has been originally attached to object-oriented concept.

Music structure (sentence, phrase, and motif) is realized by classes which refer MIDI data (precisely speaking, Psyche is extending MIDI data for its own, but "MIDI data" is used for Psyche’s MIDI data in this paper). Classes of music structures are subclasses of a class Performance in the sense that performance is affected by and realized in music structures.

Classes should have been designed that they can be commonly used by all the systems of Psyche. Psyche systems are categorized as follows.

1. visualizing music expression,
2. synthesizing music performance with rules,
3. automatic cooperative accompaniment.

As for 3, the preparation step with the human player's rehearsal is concerned. The real-time accompaniment step puts emphasis on a different software design.

The input to these systems are data of a music score, or a corresponding MIDI data file, and data which describes music structure. Psyche has music description languages for music score and for music interpretation represented by music structure. The output of the systems are MIDI data file. In other words, each system manipulates and changes the input MIDI data. This derives that one of the properties of music structure classes is referring the corresponding portion of the MIDI data.

With the design of music structure classes, at least two issues had to be solved. One is the volatility of objects and another is the handling of MIDI data in objects which consist hierarchy in music structure.

Once objects of these classes are generated, systems work on the objects. Though objects are volatile, it is desirable that objects have longer life than each process of a system because the objects represent status of the MIDI data of the time, and the next time a system might use the states. The objects are generated when the two input files are read and compiled. Especially, the file for music interpretation determines most objects’ prop-
erities. It is costing to compile files in order to generate objects each time a system runs, more over the previous states of system manipulation are lost. Also it is not an easy job to give objects which need have complicated pointers longer life than a process.

In order to solve this problem, an effect table was introduced. The effect table is made when input files are compiled. The table shows which components of a music structure has effect on other components of the same level of music structure. As there are three classes currently, three effect tables for each music structure level (motif, phrase, and sentence) are made. The table is a two-dimensional array which is easy to make it into a file (invaluable media). The next time a system starts, it is not necessary to compile input files. At closing a system, the MIDI data status is put into output data file. When objects are regenerated, the MIDI data status is derived by using the previously output MIDI data.

Components of music structure make a hierarchy. For example, a motif M1 and a motif M2 makes a phrase P1 or a sentence S1 consists of a phrase P1 and a phrase P2. As objects are generated for M1, M2, P1, and S1, MIDI data for M1 portion is referred three. Then a mechanism should be made on whether the modification on M1 in the visualization system be influenced on P1 and S1 or not. By having MIDI data only in the different parameter values from the original as its property, objects for music structure components need not to care about the consistency of the data value among objects. The output performance MIDI data is made by adding different parameter values.

5 Summary

The better usability in the visualization system is expected to activate the development of other systems of Psyche. In this paper, two types of visualization systems are introduced with their internal structure.

During over a decade of activity, a lot of systems were made for Psyche project. Psyche is on the stage of redefining its system architecture. Though all the systems mentioned in section 4 are running well, some decisions on OS and on the complete design of class libraries should be made because of the rapid appearance of more powerful hardware and software products in these days. Currently the fan-shaped expression visualization system is built on MS-DOS and the system using snake-like figure is on Unix.

Psyche will be designed into four-layer architecture (Figure 3). The effective development is expected with the architecture. As three types of

Hiraga et al. 486
ICMC Proceedings 1996

End Sex Programming

<table>
<thead>
<tr>
<th>GRID</th>
<th>Expression visualization</th>
<th>Performance synthesis</th>
<th>System integration</th>
</tr>
</thead>
<tbody>
<tr>
<td>P0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P1</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Common Library:

- effect table

- objects (static)

- objects (dynamic)

Figure 3. Psycho Software Architecture

systems refer each other more or less, a unified system design is required. In a year or so, at least a system is realized with this architecture.

References